



```
FFFFFFFFF 000000 RRRRRRRR SSSSSSSS IIIIII GGGGGGGG NN NN AAAAAA LL
FFFFFFFFF 000000 RRRRRRRR SSSSSSSS IIIIII GGGGGGGG NN NN AAAAAA LL
FF 00 00 RR RR SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR SS  II  GG  NN  NN  AA  AA  LL
FFFFFFFFF 00 00 RRRRRRRR SSSSSS  II  GG  NN  NN  AA  AA  LL
FFFFFFFFF 00 00 RRRRRRRR SSSSSS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR  SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR  SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR  SS  II  GG  NN  NN  AA  AA  LL
FF 00 00 RR RR  SS  II  GG  NN  NN  AA  AA  LL
FF 000000 RR RR SSSSSSSS IIIIII GGGGGG NN NN AA  AA  LL
FF 000000 RR RR SSSSSSSS IIIIII GGGGGG NN NN AA  AA  LL
LL  IIIII SSSSSSSS
LL  IIIII SSSSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SSSSSS
LL  II  SSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```

```
1 0001 0 MODULE FOR$$SIGNAL (%TITLE'FORTRAN SIGNAL, SIGNAL_STOP and SIG_NO_LUB'
2 0002 0 IDENT = '1-007' ! File: FORSIGNAL.B32 Edit: SBL1007
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 * ALL RIGHTS RESERVED.
12 0012 1 *
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
18 0018 1 * TRANSFERRED.
19 0019 1 *
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
22 0022 1 * CORPORATION.
23 0023 1 *
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
26 0026 1 *
27 0027 1 *
28 0028 1 *****
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: FORTRAN Support Library
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 FORTRAN support routines to convert FORTRAN error code
37 0037 1 to 32-bit VAX error code, and SIGNAL or SIGNAL_STOP
38 0038 1 extra information in format compatible for SYS$PUT_MESSAGE:
39 0039 1
40 0040 1 ENVIRONMENT: User Mode - AST re-entrant
41 0041 1 Note: this module is both shared and non-shared.
42 0042 1 If compatibility routine calls it, a non-shared copy is included.
43 0043 1 Hence, JSB to FOR$$CB_GET instead of accessing OTS$$A_CUR_LUB directly.
44 0044 1
45 0045 1 AUTHOR: Thomas N. Hastings, CREATION DATE: 8-Aug-1977
46 0046 1
47 0047 1 MODIFIED BY:
48 0048 1
49 0049 1 Thomas N. Hastings, 8-Aug-1977: VERSION 0
50 0050 1 Steven B. Lionel, VAX/VMS V2.0
51 0051 1 [Previous edit history removed. SBL 10-Nov-1980]
52 0052 1 1-001 - Update version number and copyright notice. JBS 16-NOV-78
53 0053 1 1-002 - Change LUB$B LUN to LUB$W LUN. JBS 05-DEC-78
54 0054 1 1-003 - Change REQUIRE file names from FOR... to OTS... JBS 06-DEC-78
55 0055 1 1-004 - Get filename from FAB if all else fails. SBL 29-Aug-1979
56 0056 1 1-005 - Add optional FAB argument to FOR$$SIG_NO_LUB. SBL 7-OCT-1979
57 0057 1 1-006 - Allow extra FAO arguments and conditions to be passed to
```



FOR\$\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

F 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 2  
(1)

```
: 58      0058 1 |      FOR$$SIGNAL and FOR$$SIGNAL_STO. Remove debugging macros, no longer
: 59      0059 1 |      used. SBL 10-Nov-1980
: 60      0060 1 |      *** First post-V3.0 edit ***
: 61      0061 1 |      1-007 - Use prologue file. SBL 20-Jan-1983
: 62      0062 1 |      --
```

```

64      0063 1 |
65      0064 1 | PROLOGUE FILE:
66      0065 1 |
67      0066 1 |
68      0067 1 | REQUIRE 'RTLIN:FORPROLOG';          ! FOR$ Definitions
69      0133 1 |
70      0134 1 |
71      0135 1 | TABLE OF CONTENTS:
72      0136 1 |
73      0137 1 |
74      0138 1 | FORWARD ROUTINE
75      0139 1 |     FOR$$SIGNAL: NOVALUE,          ! SIGNAL 32-bit error code and LUB data
76      0140 1 |     FOR$$SIGNAL_STO: NOVALUE,      ! SIGNAL_STOP 32-bit error code and LUB data
77      0141 1 |     FOR$$SIG_FATINT: NOVALUE,      ! SIGNAL_STOP OT$$ FATINTERR (FATAL
78      0142 1 |                                     ! INTERNAL ERROR IN RUN-TIME LIBRARY)
79      0143 1 |     FOR$$SIG_DATCOR: NOVALUE,      ! SIGNAL_STOP OT$$ INTDATCOR (INTERNAL
80      0144 1 |                                     ! DATA CORRUPTED IN RUN-TIME LIBRARY)
81      0145 1 |     DO SIGNAL: NOVALUE,            ! Do the work for FOR$$SIGNAL, FOR$$SIGNAL_STO
82      0146 1 |     FOR$$SIG_NO_LUB: NOVALUE,      ! SIGNAL_STOP with no LUB setup.
83      0147 1 |     COND_VALUE;                   ! Return 32-bit condition value given FORTRAN error #
84      0148 1 |
85      0149 1 |
86      0150 1 | MACROS:
87      0151 1 |
88      0152 1 |
89      0153 1 |
90      0154 1 | EQUATED SYMBOLS:
91      0155 1 |
92      0156 1 | LITERAL
93      0157 1 |     K_NO_FAO_SIGARG = 3;          ! No. of FAO args in signal arg list
94      0158 1 |                                     ! used by SY$$PUT_MESSAGE
95      0159 1 |
96      0160 1 | OWN STORAGE:
97      0161 1 |
98      0162 1 |     NONE
99      0163 1 |
100     0164 1 |
101     0165 1 | EXTERNAL REFERENCES:
102     0166 1 |
103     0167 1 |
104     0168 1 | +
105     0169 1 | MAINTENANCE NOTE: Since this module is called by FORTRAN compatibility
106     0170 1 | routines which are un-shared and the entry points are not vectored,
107     0171 1 | a separate copy of this module is linked with the user program when
108     0172 1 | the user calls a FORTRAN compatibility routine. In order to prevent
109     0173 1 | data truncation errors from the linker, all external references are
110     0174 1 | of addressing mode general (rather than word displacement) even for
111     0175 1 | the same PSECT.
112     0176 1 | -
113     0177 1 |
114     0178 1 |
115     0179 1 | EXTERNAL ROUTINE
116     0180 1 |     FOR$$CB GET: JSB_CB GET NOVALUE ADDRESSING MODE (GENERAL),      ! Set CCB to adr. of current LUB/ISB
117     0181 1 |     FOR$$ERRSNS SAV: NOVALUE ADDRESSING MODE (GENERAL),             ! Save error info for FOR$ERRSNS.
118     0182 1 |     LIB$$SIGNAL: NOVALUE ADDRESSING MODE (GENERAL),                 ! SIGNAL error and continue
119     0183 1 |     LIB$$STOP: NOVALUE ADDRESSING_MODE (GENERAL);                   ! SIGNAL error and STOP
120     0184 1 |
```

FORSSIGNAL  
1-007

FORTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

H 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 4  
(2)

:	121	0185	1	EXTERNAL LITERAL		
:	122	0186	1		OTSS_FATINTERR: UNSIGNED (%BPVAL),	! Condition value FATAL INTERNAL ERROR IN RUN-TIME LIBRARY
:	123	0187	1		OTSS_INTDATCOR: UNSIGNED (%BPVAL);	! Condition value INTERNAL DATA CORRUPTED IN RUN-TIME LIBRAR
:	124	0188	1			



```
126 0189 1 GLOBAL ROUTINE FOR$$SIGNAL      ! SIGNAL FORTRAN error and continue
127 0190 1      :NOVALUE =                  ! No value returned.
128 0191 1
129 0192 1 ++
130 0193 1 FUNCTIONAL DESCRIPTION:
131 0194 1      Signals a FORTRAN-specific error whose 32-bit condition code or
132 0195 1      small-integer error number is the first argument.  If other arguments
133 0196 1      are present, they represent extra FAO arguments for the first
134 0197 1      condition and/or secondary conditions to be signalled.  See DO_SIGNAL
135 0198 1      for more information
136 0199 1
137 0200 1 CALLING SEQUENCE:
138 0201 1
139 0202 1      CALL FOR$$SIGNAL (fort_err_no.rc.v [,fao_args_0.rz.v, ...]
140 0203 1      [,secondary_msg.rc.v [,sec_fao_cnt.rl.v[, sec_fao_args.rz.v,...]])
141 0204 1
142 0205 1 FORMAL PARAMETERS:
143 0206 1
144 0207 1      fort_err_no      - A 32-bit FOR$ code or the small integer which is the
145 0208 1                      error number part of a FOR$ code.
146 0209 1      fao_args_0        - FAO arguments for this message.  The three FAO arguments
147 0210 1                      unit number, filename and user PC are always used; if
148 0211 1                      fao_args_0 are specified, they come before the default
149 0212 1                      arguments.
150 0213 1      secondary_msg     - Secondary message to be signalled.  MUST be a 32-bit code.
151 0214 1      sec_fao_cnt       - FAO count for secondary message
152 0215 1      sec_fao_args      - FAO arguments for secondary message
153 0216 1
154 0217 1 IMPLICIT INPUTS:
155 0218 1
156 0219 1      See DO_SIGNAL
157 0220 1
158 0221 1 IMPLICIT OUTPUTS:
159 0222 1
160 0223 1      See DO_SIGNAL
161 0224 1
162 0225 1 COMPLETION CODES:
163 0226 1
164 0227 1      NONE
165 0228 1
166 0229 1 SIDE EFFECTS:
167 0230 1
168 0231 1      Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNALs.
169 0232 1      Saves error info in FOR$ERRSNS OWN storage.
170 0233 1
171 0234 1 --
172 0235 2 BEGIN
173 0236 2
174 0237 2 BUILTIN
175 0238 2 AP;
176 0239 2
177 0240 2 DO_SIGNAL (.AP, LIB$$SIGNAL);
178 0241 2
179 0242 2 RETURN
180 0243 1 END;
```

FOR\$\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

J 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 6  
(3)

;

.TITLE FOR\$\$SIGNAL FORTTRAN SIGNAL, SIGNAL\_STOP and SIG  
\_NO\_LUB

.IDENT \1-007\

.EXTRN FOR\$\$CB GET, FOR\$\$ERRSNS\_SAV

.EXTRN LIB\$\$SIGNAL, LIB\$\$STOP

.EXTRN OT\$\$\_FATINTERR, OT\$\$\_INTDATCOR

.PSECT \_FOR\$CODE, NOWRT, SHR, PIC, 2

.ENTRY FOR\$\$SIGNAL, Save nothing

PUSHAB LIB\$\$SIGNAL

PUSHL AP

CALLS #2, DO\_SIGNAL

RET

: 0189

: 0240

: 0243

00000000G 00 0000 00000  
5C 9F 00002  
0000V CF 02 DD 00008  
04 FB 0000A  
04 0000F

; Routine Size: 16 bytes, Routine Base: \_FOR\$CODE + 0000



FOR\$\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

K 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 7  
(4)

```
: 182      0244 1 GLOBAL ROUTINE FOR$$SIGNAL_STO  ! SIGNAL_STOP FORTTRAN error and STOP
: 183      0245 1      :NOVALUE =                ! No value returned.
: 184      0246 1      ++
: 185      0247 1      FUNCTIONAL DESCRIPTION:
: 186      0248 1
: 187      0249 1      Convert FORTTRAN error code number to 32-bit VAX-11 error code.
: 188      0250 1      See description for FOR$$SIGNAL above which is identical,
: 189      0251 1      except that FOR$$SIGNAL_STO calls LIB$STOP instead of LIB$SIGNAL.
: 190      0252 1      --
: 191      0253 1
: 192      0254 2      BEGIN
: 193      0255 2
: 194      0256 2      BUILTIN
: 195      0257 2      AP;
: 196      0258 2
: 197      0259 2      DO_SIGNAL (.AP, LIB$STOP);
: 198      0260 2
: 199      0261 2      RETURN
: 200      0262 1      END;
```

```
00000000G 00 0000 0000
          5C 9F 0002
0000V CF  02 DD 0008
          04 FB 000A
          04 000F
```

```
.ENTRY FOR$$SIGNAL_STO, Save nothing
PUSHAB LIB$STOP
PUSHL AP
CALLS #2, DO_SIGNAL
RET
```

```
: 0244
: 0259
:
: 0262
```

; Routine Size: 16 bytes, Routine Base: \_FOR\$CODE + 0010

FOR\$\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

L 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 8  
(5)

```
: 202      0263 1 GLOBAL ROUTINE FOR$$SIG_FATINT      ! SIGNAL_STOP OTSS_FATINTERR and STOP
: 203      0264 1      :NOVALUE =                  ! No value returned.
: 204      0265 1      ++
: 205      0266 1      FUNCTIONAL DESCRIPTION:
: 206      0267 1
: 207      0268 1      SIGNAL_STOP OTSS_FATINTERR = FATAL INTERNAL ERROR IN RUN-TIME LIBRARY.
: 208      0269 1      Note: The current LUB (if any) is ignored and no UNIT is printed.
: 209      0270 1      --
: 210      0271 1
: 211      0272 2      BEGIN
: 212      0273 2      FOR$$SIG_NO_LUB (OTSS_FATINTERR);
: 213      0274 2      RETURN
: 214      0275 1      END;
```

```
0000V CF 00000000G 8F DD 00002
01 FB 00008
04 0000D
```

```
.ENTRY FOR$$SIG_FATINT, Save nothing
PUSHL #OTSS_FATINTERR
CALLS #1, FOR$$SIG_NO_LUB
RET
```

```
: 0263
: 0273
: 0275
```

; Routine Size: 14 bytes, Routine Base: \_FOR\$CODE + 0020

FOR\$\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

M 16  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 7  
(6)

```
: 216      0276 1 GLOBAL ROUTINE FOR$$SIG_DATCOR      ! SIGNAL_STOP OTSS_INTDATCOR and STOP
: 217      0277 1      :NOVALUE =                  ! No value returned.
: 218      0278 1      ++
: 219      0279 1      FUNCTIONAL DESCRIPTION:
: 220      0280 1
: 221      0281 1      SIGNAL_STOP OTSS_INTDATCOR = INTERNAL DATA CORRUPTED IN RUN-TIME LIBRARY.
: 222      0282 1      Note: The current LUB (if any) is ignored and no UNIT is printed.
: 223      0283 1      --
: 224      0284 1
: 225      0285 2      BEGIN
: 226      0286 2      FOR$$SIG_NO_LUB (OTSS_INTDATCOR);
: 227      0287 2      RETURN
: 228      0288 1      END;
```

```
0000V CF 00000000G 8F DD 00002
01 FB 00008
04 0000D
```

```
.ENTRY FOR$$SIG_DATCOR, Save nothing
PUSHL #OTSS_INTDATCOR
CALLS #1, FOR$$SIG_NO_LUB
RET
```

```
: 0276
: 0286
: 0288
```

; Routine Size: 14 bytes, Routine Base: \_FOR\$CODE + 002E



```
230 0289 1 ROUTINE DO SIGNAL (      ! Internal routine to do work for FOR$$SIGNAL and FOR$$SIGNAL_STO
231 0290 1     SIGNAL_LIST_ARG,      ! List of arguments to signal routine
232 0291 1     SIGNAL_ROUTINE)      ! adr. of LIB$$SIGNAL or LIB$$STOP
233 0292 1     : NOVALUE =          ! No value returned
234 0293 1
235 0294 1 ++
236 0295 1 FUNCTIONAL DESCRIPTION:
237 0296 1     Converts error code number to 32-bit VAX-11 error code.
238 0297 1     See description of FOR$$SIGNAL above.
239 0298 1
240 0299 1 FORMAL PARAMETERS:
241 0300 1
242 0301 1     SIGNAL_LIST_ARG      Contents of AP at time of call to FOR$$SIGNAL or
243 0302 1                       FOR$$SIGNAL STO
244 0303 1     SIGNAL_ROUTINE      ADR. of LIB$$SIGNAL or LIB$$STOP
245 0304 1
246 0305 1 IMPLICIT INPUTS:
247 0306 1
248 0307 1     OTSS$A_CUR_LUB          ADR. of current LUB/ISB/RAB
249 0308 1                       Obtained by JSB to FOR$$CB_GET.
250 0309 1     {FAB,RAB}$L_STS      RMS error status
251 0310 1     {FAB,RAB}$L_STV      RMS error value or operating system error code
252 0311 1
253 0312 1 IMPLICIT OUTPUTS:
254 0313 1
255 0314 1     {FAB,RAB}$L_STS          RMS error status - set to 0
256 0315 1     {FAB,RAB}$L_STV      RMS error value or operating system error code - set to 0
257 0316 1     FORTRAN error #, RMS STS, RMS STV, logical unit number saved in
258 0317 1                       OWN storage in FOR$ERRSNS module for later
259 0318 1                       call by user to ERRSNS.
260 0319 1
261 0320 1 COMPLETION CODES:
262 0321 1
263 0322 1     NONE
264 0323 1
265 0324 1 SIDE EFFECTS:
266 0325 1
267 0326 1     Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNALS.
268 0327 1     Saves error info in FOR$ERRSNS OWN storage.
269 0328 1
270 0329 1
```

```
272 0330 BEGIN
273 0331
274 0332 GLOBAL REGISTER
275 0333 CCB = K_CCB_REG: REF $FOR$CCB_DECL;
276 0334
277 0335 LOCAL
278 0336 FILE_NAME_DSC: DSC$DESCRIPTOR, ! File name descriptor for resultant file name
279 0337 RABORFAB: REF BLOCK[, BYTE],
280 0338 STS, ! RMS RAB or FAB error status
281 0339 STV, ! RMS RAB or FAB error status
282 0340 GETMSG_VALS: VECTOR [4, BYTE], ! Returned values from $GETMSG
283 0341 SIGNAL_LIST: VECTOR [20, LONG], ! Argument list to LIB$SIGNAL/LIB$STOP
284 0342 LIST_PTR: REF VECTOR [, LONG], ! Pointer into signal list
285 0343 ARGS_PTR: REF VECTOR [, LONG], ! pointer into SIGNAL_LIST_ARG
286 0344 ARG [1:END, ! Address of argument list end
287 0345 COND_VAL: BLOCK [4, BYTE]; ! 32-bit VAX-11 error code
288 0346
289 0347 MAP
290 0348 SIGNAL_LIST_ARG: REF VECTOR [, LONG];
291 0349
292 0350 BUILTIN
293 0351 CALLG;
294 0352
295 0353 FOR$CB_GET (); ! Set CCB to adr. of current LUB/ISB/RAB
296 0354
297 0355 !+
298 0356 Convert FORTRAN error code to 32-bit VAX-11 error code.
299 0357 Conversion is done by copying FORTRAN error number to code field,
300 0358 setting the severity code to SEVERE,
301 0359 for all errors except FOR$OUTCONERR (63='OUTPUT CONVERSION ERROR')
302 0360 which is set to ERROR instead so that image will continue
303 0361 by default since output field is flagged with ***s.
304 0362 All other continuable errors are signaled SEVERE so that user
305 0363 must take overt action in order to continue past the error.
306 0364 setting the facility code to FOR$K FAC_NO,
307 0365 and setting the facility specific Bit (STS$V_FAC_SP).
308 0366 !-
309 0367
310 0368 COND_VAL = COND_VALUE (.SIGNAL_LIST_ARG [1]);
311 0369
312 0370 !+
313 0371 Call $GETMSG to see how many FAC parameters it takes.
314 0372 !-
315 0373
316 0374 BEGIN
317 0375 LOCAL
318 0376 DSC: VECTOR [2, LONG],
319 0377 LEN;
320 0378 DSC [0] = 0; ! Null string descriptor
321 0379 DSC [1] = LEN;
322 0380 $GETMSG (
323 0381 MSGID = .COND_VAL,
324 0382 MSGLEN = LEN,
325 0383 BUFADR = DSC,
326 0384 FLAGS = 0,
327 0385 OUTADR = GETMSG_VALS);
328 0386 END;
```

```
329 0387 2
330 0388 2
331 0389 2
332 0390 2
333 0391 2
334 0392 2
335 0393 2
336 0394 2
337 0395 2
338 0396 2
339 0397 2
340 0398 2
341 0399 2
342 0400 2
343 0401 2
344 0402 2
345 0403 2
346 0404 2
347 0405 2
348 0406 2
349 0407 2
350 0408 2
351 0409 2
352 0410 2
353 0411 2
354 0412 2
355 0413 2
356 0414 2
357 0415 2
358 0416 2
359 0417 2
360 0418 2
361 0419 2
362 0420 2
363 0421 2
364 0422 2
365 0423 2
366 0424 2
367 0425 2
368 0426 2
369 0427 2
370 0428 2
371 0429 2
372 0430 2
373 0431 2
374 0432 2
375 0433 2
376 0434 2
377 0435 2
378 0436 2
379 0437 2
380 0438 2
381 0439 2
382 0440 2
383 0441 2
384 0442 2
385 0443 2

!+
!- Compute total number of signal arguments.
SIGNAL_LIST [0] = (.SIGNAL_LIST_ARG [0])<0,8,0> + 6;
ARG_LIST_END = SIGNAL_LIST_ARG [0] + ((.SIGNAL_LIST_ARG [0])<0,8,0> * %UPVAL);

!+
!- Fill in primary condition message.
SIGNAL_LIST [1] = .COND_VAL;
SIGNAL_LIST [2] = .GETMSG_VALS [1]; ! Number of FAO parameters
LIST_PTR = SIGNAL_LIST [3];
ARGS_PTR = SIGNAL_LIST_ARG [2];

!+
!- Copy extra FAO arguments, if any.
INCR I FROM 4 TO .SIGNAL_LIST [2] DO
    COPY_LONG_A (ARGS_PTR, LIST_PTR);

!+
!- Get RMS error status from RAB or if not error there from FAB (if any).
!- Then set error status longwords to 0 so will not be found again.
!- Note: this code depends on the fact that FAB$L_STS/STV have the same offsets
!- as RAB$L_STS/STV.
STS = 0; ! Set initial values
STV = 0;
IF .CCB [LUB$W_LUN] NEQU LUB$K_LUN_ENCD ! Not ENCODE/DECODE/internal?
THEN
    BEGIN
        RABORFAB = .CCB;
        IF (.CCB[RAB$L_STS] OR .CCB[RAB$L_STS] EQL 0)
        THEN
            RABORFAB = .CCB[RAB$L_FAB];

        IF NOT .RABORFAB[RAB$L_STS]
        THEN
            BEGIN
                STS = .RABORFAB[RAB$L_STS];
                STV = .RABORFAB[RAB$L_STV];
            END;

        RABORFAB[RAB$L_STS] = 0;
        RABORFAB[RAB$L_STV] = 0;
    END;

!+
!- Save FORTRAN error number, RMS STS, RMS STV, logical unit number and VAX-11 condition value
```



```
386 0444 2 FOR$$ERRSNS_SAV (.COND_VAL [STSV_CODE], .STS, .STV, .CCB[LUB$W_LUN], .COND_VAL);
387 0445 2
388 0446 2
389 0447 2
390 0448 2
391 0449 2
392 0450 2
393 0451 2
394 0452 2
395 0453 2
396 0454 2
397 0455 2
398 0456 2
399 0457 2
400 0458 2
401 0459 2
402 0460 2
403 0461 2
404 0462 2
405 0463 2
406 0464 2
407 0465 2
408 0466 2
409 0467 2
410 0468 2
411 0469 2
412 0470 2
413 0471 2
414 0472 2
415 0473 2
416 0474 2
417 0475 2
418 0476 2
419 0477 2
420 0478 2
421 0479 2
422 0480 2
423 0481 2
424 0482 2
425 0483 2
426 0484 2
427 0485 1
```

```
FOR$$ERRSNS_SAV (.COND_VAL [STSV_CODE], .STS, .STV, .CCB[LUB$W_LUN], .COND_VAL);
!+
! Set up resultant file name descriptor that gets put in signal arg list.
! Note that this points at the FAB's FNM until the file is opened.
!-
FILE_NAME_DSC[DSC$W_LENGTH] = .CCB[LUB$B_RSL];
FILE_NAME_DSC[DSC$B_DTYPE] = FILE_NAME_DSC[DSC$B_CLASS] = 0;
FILE_NAME_DSC[DSC$A_POINTER] =
    IF .CCB[LUB$B_RSL] EQLU 0
    THEN
        0
    ELSE
        .CCB[LUB$A_RSN]);
!+
! Insert the three default FAO arguments plus the STS and STV.
!-
LIST_PTR [0] = .CCB [LUB$W_LUN];
LIST_PTR [1] = FILE_NAME_DSC;
LIST_PTR [2] = 0; ! For user PC
LIST_PTR [3] = .STS;
LIST_PTR [4] = .STV;
LIST_PTR = LIST_PTR [5];
WHILE .ARGS_PTR LEQ .ARG_LIST_END DO
    COPY_LONG_A (ARGS_PTR, LIST_PTR);
!+
! Call LIB$STOP to STOP the error or LIB$SIGNAL to SIGNAL the error.
!-
CALLG (SIGNAL_LIST, .SIGNAL_ROUTINE);
!+
! Return
!-
RETURN
END; ! End of FOR$$SIGNAL_STO routine
```

.EXTRN SYS\$GETMSG

08FC 0000 DO\_SIGNAL:

SE	98	AE	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R11	: 0289
	000000006	00	16	00006	MOVAB	-104(SP), SP	: 0353
52	04	AC	D0	0000C	JSB	FOR\$\$CB_GET	: 0368
	04	A2	DD	00010	MOVL	SIGNAL_LIST_ARG, R2	
0000V	CF	01	FB	00013	PUSHL	4(R2)	
	54	50	D0	00018	CALLS	#1, COND_VALUE	
		AE	D4	0001B	MOVL	R0, COND_VAL	
OC	AE	6E	9E	0001E	CLRL	DSC	: 0378
					MOVAB	LEN, DSC+4	: 0379

		04	AE	9F	00022	PUSHAB	GETMSG_VALS	0385
			7E	D4	00025	CLRL	-(SP)	
		10	AE	9F	00027	PUSHAB	DSC	
		0C	AE	9F	0002A	PUSHAB	LEN	
			54	DD	0002D	PUSHL	COND_VAL	
00000000G	00		05	FB	0002F	CALLS	#5, SYS\$GETMSG	
	50		62	D0	00036	MOVL	(R2), R0	0392
	10	AE	06	A0	9E	MOVAB	6(R0), SIGNAL_LIST	
	57		8240	DE	0003E	MOVAL	(R2)+[R0], ARG_LIST_END	0393
	14	AE		54	D0	MOVL	COND_VAL, SIGNAL_LIST+4	0399
	18	AE	05	AE	9A	MOVZBL	GETMSG_VALS+1, SIGNAL_LIST+8	0400
		53	1C	AE	9E	MOVAB	SIGNAL_LIST+12, LIST_PTR	0402
		52		04	C0	ADDL2	#4, ARGS_PTR	0403
		50		03	D0	MOVL	#3, I	0409
			03	11	00055	BRB	2\$	
	83		82	D0	00057	1\$: MOVL	(ARGS_PTR)+, (LIST_PTR)+	0410
F8	50		18	AE	F3	2\$: AOBLEQ	SIGNAL_LIST+8, I, TS	
				55	7C		STV	0420
	FFFB	8F	C6	AB	B1	CLRQ	-58(CCB), #-5	0421
				1F	13	CMPL	6\$	
		50		5B	D0	BEQL	CCB, RABORFAB	0424
		05	08	AB	E8	MOVL	8(CCB), 3\$	0425
			08	AB	D5	BLBS	8(CCB)	
				04	12	TSTL	4\$	
		50	3C	AB	D0	BNEQ	60(CCB), RABORFAB	0427
		08	08	A0	E8	MOVL	8(RABORFAB), 5\$	0429
		56	08	A0	D0	BLBS	8(RABORFAB), STS	0432
		55	0C	A0	D0	MOVL	12(RABORFAB), STV	0433
			08	A0	7C	MOVL	8(RABORFAB)	0436
				54	DD	CLRQ	COND_VAL	0444
		7E	C6	AB	32	PUSHL	-58(CCB), -(SP)	
				55	DD	CVTWL	STV	
				56	DD	PUSHL	STS	
7E	54	0C		03	EF	PUSHL	#3, #12, COND_VAL, -(SP)	
	00000000G	00		05	FB	EXTZV	#5, FOR\$ERRSNS SAV	
	60	AE	F7	AB	9B	CALLS	-9(CCB), FILE_NAME_DSC	0451
			62	AE	B4	MOVZBW	FILE_NAME_DSC+2	0452
			F7	AB	95	CLRW	-9(CCB)	0454
				04	12	TSTB	7\$	
				50	D4	BNEQ	R0	
				04	11	CLRL	8\$	
		50	F8	AB	D0	BRB	-8(CCB), R0	0458
	64	AE		50	D0	MOVL	R0, FILE_NAME_DSC+4	0454
		83	C6	AB	32	MOVL	-58(CCB), (LIST_PTR)+	0464
		83	60	AE	9E	CVTWL	FILE_NAME_DSC, (LIST_PTR)+	0465
				83	D4	MOVAB	(LIST_PTR)+	0466
		83		56	D0	CLRL	STS, (LIST_PTR)+	0467
		83		55	D0	MOVL	STV, (LIST_PTR)+	0468
		57		52	D1	MOVL	ARGS_PTR, ARG_LIST_END	0471
				05	14	CMPL	10\$	
		83		82	D0	BGTR	(ARGS_PTR)+, (LIST_PTR)+	0472
				F6	11	MOVL	9\$	
	08	BC	10	AE	FA	BRB	SIGNAL_LIST, @SIGNAL_ROUTINE	0478
				04	000D6	CALLG		0485
						RET		

; Routine Size: 215 bytes, Routine Base: \_FOR\$CODE + 003C

FORSSIGNAL  
1-007

FORTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

<sup>6</sup><sub>1</sub>  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 15  
(8)

FO  
1-



```
429 0486 1 GLOBAL ROUTINE FOR$$SIG_NO_LUB ( ! SIGNAL_STOP FORTRAN error and STOP
430 0487 1   FORT_ERR_NO, ! FORTRAN error code 0:120 or 32-bit cond value
431 0488 1   ! VAX-11 error code
432 0489 1   FORT_LUN, ! Optional FORTRAN logical unit number
433 0490 1   FAB) ! Optional FAB address
434 0491 1   :NOVALUE = ! No value returned.
435 0492 1
436 0493 1 ++
437 0494 1 FUNCTIONAL DESCRIPTION:
438 0495 1   Convert FORTRAN error code number to 32-bit VAX-11 error code.
439 0496 1   The following SIGNAL_STOP arguments are obtained from the
440 0497 1   argument list only since no LUB/ISB/RAB yet:
441 0498 1
442 0499 1   VAX-11 error code:
443 0500 1       STSSV_SEVERITY = STSK_SEVERE
444 0501 1       STSSV_CODE = FORTRAN error number
445 0502 1       STSSV_FAC_SP = 1 (facility specific error messages
446 0503 1       STSSV_FAC_NO = FORTRAN facility no. (FOR$K_FAC_NO)
447 0504 1   3 = No. of following FAO arguments
448 0505 1   FORTRAN unit number if present or zero
449 0506 1   File name string descriptor address or 0 if no FAB
450 0507 1   User PC of call to library (set to 0 here, rewritten by handler before RESIGNAL)
451 0508 1   RMS error code from FAB if present
452 0509 1   System error code from FAB if present
453 0510 1
454 0511 1 FORMAL PARAMETERS:
455 0512 1
456 0513 1   FORT_ERR_NO.rlu.v   FORTRAN error code (0:120) or 32-bit cond value
457 0514 1   [FORT_LUN.rlu.v]   32-bit VAX-11 error code with LH already set.
458 0515 1   [FAB.rbu.ra]       Optional unit number, 0 used if not present
459 0516 1   Address of FAB if present
460 0517 1
461 0518 1 IMPLICIT INPUTS:
462 0519 1
463 0520 1   NONE
464 0521 1
465 0522 1 IMPLICIT OUTPUTS:
466 0523 1
467 0524 1   NONE
468 0525 1
469 0526 1 COMPLETION CODES:
470 0527 1
471 0528 1   NONE
472 0529 1
473 0530 1 SIDE EFFECTS:
474 0531 1
475 0532 1   Converts FORTRAN error code to 32-bit VAX-11 error code and SIGNAL_STOPs.
476 0533 1 --
```

```
478 0534 BEGIN
479 0535 LOCAL
480 0536     VAX_11_COND_VAL: BLOCK[4,BYTE], | 32-bit VAX-11 error code
481 0537     NAME_DSC : DSC$DESCRIPTOR,      | File name descriptor
482 0538     STS,      | RMS error status
483 0539     STV;      | System error status
484 0540
485 0541 MAP
486 0542     FORT_ERR_NO: BLOCK[,BYTE],      | MAKE 32-bit VAX-11 error code
487 0543     FAB : REF BLOCK [,BYTE];      | FAB is address of FAB
488 0544
489 0545 BUILTIN
490 0546     ACTUALCOUNT;                  | Actual no. of parameters
491 0547
492 0548 | +
493 0549 | Convert FORTRAN error code to 32-bit VAX-11 error code unless
494 0550 | already converted by the caller. Conversion is done
495 0551 | by copying FORTRAN error number to code field,
496 0552 | setting the severity code to SEVERE,
497 0553 | setting the facility code to FOR$K FAC_NO,
498 0554 | and setting the facility specific Bit (STS$V_FAC_SP).
499 0555 | -
500 0556 VAX_11_COND_VAL = COND_VALUE (.FORT_ERR_NO);
501 0557
502 0558 | +
503 0559 | If FAB argument is present, retrieve RMS and SYSTEM error codes.
504 0560 | -
505 0561 IF ACTUALCOUNT () GTRU 2
506 0562 THEN
507 0563     BEGIN
508 0564         STS = (IF .FAB [FAB$STV] THEN 0 ELSE .FAB [FAB$STV]);
509 0565         STV = (IF .FAB [FAB$STV] THEN 0 ELSE .FAB [FAB$STV]);
510 0566     END
511 0567 ELSE
512 0568     BEGIN
513 0569         STS = 0;
514 0570         STV = 0;
515 0571     END;
516 0572
517 0573 | +
518 0574 | Save FORTRAN error #, RMS STS, RMS STV, logical unit number, and VAX-11 condition value.
519 0575 | If FORT_LUN not present, use 0 (e.g., INVALID ARG TO FORTRAN I/O SYSTEM)
520 0576 | -
521 0577
522 0578 FOR$ERRSNS SAV (.FORT_ERR_NO, .STS, .STV,
523 0579     (IF ACTUALCOUNT () GTRU 1 THEN .FORT_LUN ELSE 0), .VAX_11_COND_VAL);
524 0580
525 0581 | +
526 0582 | Set up file name descriptor
527 0583 | -
528 0584 NAME_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
529 0585 NAME_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
530 0586 IF ACTUALCOUNT () GTRU 2
531 0587 THEN
532 0588     BEGIN
533 0589         IF .FAB [FAB$NAM] NEQ 0
534 0590         THEN
```

```

0591 4      BEGIN
0592 4      LOCAL
0593 4      NAM : REF BLOCK [BYTE];      ! NAM block
0594 4      NAM = .FAB [FAB$L_NAM];
0595 4      IF .NAM [NAM$B_RSC] NEQ 0
0596 4      THEN
0597 4      BEGIN
0598 4      NAME_DSC [DSC$W_LENGTH] = .NAM [NAM$B_RSL];
0599 4      NAME_DSC [DSC$A_POINTER] = .NAM [NAM$C_RSA];
0600 4      END
0601 4      ELSE IF .NAM [NAM$B_ESL] NEQ 0
0602 4      THEN
0603 4      BEGIN
0604 4      NAME_DSC [DSC$W_LENGTH] = .NAM [NAM$B_ESL];
0605 4      NAME_DSC [DSC$A_POINTER] = .NAM [NAM$C_ESA];
0606 4      END
0607 4      ELSE
0608 4      BEGIN
0609 4      NAME_DSC [DSC$W_LENGTH] = .FAB [FAB$B_FNS];
0610 4      NAME_DSC [DSC$A_POINTER] = .FAB [FAB$C_FNA];
0611 4      END;
0612 4      END
0613 4      ELSE
0614 4      BEGIN
0615 4      NAME_DSC [DSC$W_LENGTH] = .FAB [FAB$B_FNS];
0616 4      NAME_DSC [DSC$A_POINTER] = .FAB [FAB$C_FNA];
0617 4      END;
0618 4      END
0619 4      ELSE
0620 4      BEGIN
0621 4      NAME_DSC [DSC$W_LENGTH] = 0;
0622 4      NAME_DSC [DSC$A_POINTER] = 0;
0623 4      END;
0624 4
0625 4      !+
0626 4      ! call LIB$STOP to SIGNAL_STOP the error
0627 4      ! Order of args is same as defined in FPAR.MDL for use with SYS$PUT_MESSAGE
0628 4      !-
0629 4
0630 4      LIB$STOP (
0631 4      .VAX_11_COND_VAL,      ! 32-bit VAX-11 error code
0632 4      K_NO_FAB_SIGARG,      ! no. of FAB arguments following in FORTRAN error message
0633 4      .FORT_LUN,           ! FORTRAN logical unit number
0634 4      NAME_DSC,            ! file name descriptor
0635 4      0,                   ! Leave room for user PC to be filled in
0636 4      .STS,                ! by FORTRAN specific handler established on user call
0637 4      .STV);               ! RMS error code
0638 4
0639 4      ! SYSTEM error code
0640 4
0641 4      !+
0642 4      ! Return
0643 4      !-
0644 4
0645 4      RETURN
0645 4      ! End of FOR$$SIG_NO_LUB routine
0645 4      END;
```



			001C	00000	.ENTRY	FORSSIG_NO_LUB, Save R2,R3,R4	0486
	5E		08	C2	SUBL2	#8, SP	
		04	AC	DD	PUSHL	FORT_ERR_NO	0555
0000V	CF		01	FB	CALLS	#1, COND_VALUE	
	54		50	D0	MOVL	R0, VAX_T1_COND_VAL	
	02		6C	91	CMPB	(AP), #2	0561
			1A	1B	BLEQU	3\$	
	50	0C	AC	D0	MOVL	FAB, R0	0564
	04	08	A0	E9	BLBC	8(R0), 1\$	
			53	D4	CLRL	STS	
			04	11	BRB	2\$	
	53	08	A0	D0	MOVL	8(R0), STS	
	08	0C	A0	E8	BLBS	12(R0), 4\$	0565
	52	0C	A0	D0	MOVL	12(R0), STV	
			04	11	BRB	5\$	0561
			53	D4	CLRL	STS	0569
			52	D4	CLRL	STV	0570
			54	DD	PUSHL	VAX_11_COND_VAL	0579
	01		6C	91	CMPB	(AP), #1	
			05	1B	BLEQU	6\$	
		08	AC	DD	PUSHL	FORT_LUN	
			02	11	BRB	7\$	
			7E	D4	CLRL	-(SP)	
			52	DD	PUSHL	STV	0578
			53	DD	PUSHL	STS	
		04	AC	DD	PUSHL	FORT_ERR_NO	
00000000G	00		05	FB	CALLS	#5, FORSSERRSNS_SAV	0585
	02	010E	8F	B0	MOVW	#270, NAME_DSC+2	0586
			6C	91	CMPB	(AP), #2	
			38	1B	BLEQU	10\$	
	51	0C	AC	D0	MOVL	FAB, R1	0589
		28	A1	D5	TSTL	40(R1)	
			24	13	BEQL	9\$	
	50	28	A1	D0	MOVL	40(R1), NAM	0594
		03	A0	95	TSTB	3(NAM)	0595
			0B	13	BEQL	8\$	
	6E	03	A0	9B	MOVZBW	3(NAM), NAME_DSC	0598
04	AE	04	A0	D0	MOVL	4(NAM), NAME_DSC+4	0599
			20	11	BRB	11\$	0595
		0B	A0	95	TSTB	11(NAM)	0601
			0B	13	BEQL	9\$	
	6E	0B	A0	9B	MOVZBW	11(NAM), NAME_DSC	0604
04	AE	0C	A0	D0	MOVL	12(NAM), NAME_DSC+4	0605
			10	11	BRB	11\$	0601
	6E	34	A1	9B	MOVZBW	52(R1), NAME_DSC	0615
04	AE	2C	A1	D0	MOVL	44(R1), NAME_DSC+4	0616
			05	11	BRB	11\$	0586
			6E	B4	CLRW	NAME_DSC	0621
		04	AE	D4	CLRL	NAME_DSC+4	0622
			52	DD	PUSHL	STV	0638
			53	DD	PUSHL	STS	0637
			7E	D4	CLRL	-(SP)	0630
		0C	AE	9F	PUSHAB	NAME_DSC	
		08	AC	DD	PUSHL	FORT_LUN	0633

FOR\$SIGNAL  
1-007

FORTTRAN SIGNAL, SIGNAL\_STOP and SIG\_NO\_LUB

L 1  
16-Sep-1984 00:44:51  
14-Sep-1984 12:32:44

VAX-11 Bliss-32 V4.0-742  
[FORRTL.SRC]FORSIGNAL.B32;1

Page 20  
(10)

00000000G 00

03 DD 000A3  
54 DD 000A5  
07 FB 000A7  
04 000AE

PUSHL #3  
PUSHL VAX\_11 COND VAL  
CALLS #7,-LIB\$STOP  
RET

: 0630  
: 0631  
: 0645

; Routine Size: 175 bytes, Routine Base: \_FOR\$CODE + 0113

```
591 0646 1 ROUTINE COND_VALUE ( ! Internal routine to convert from FORTRAN error #
592 0647 1 ! to VAX-11 condition value
593 0648 1 ! Value of FORTRAN error # (0:120) or 32-bit cond value
594 0649 1 ! Value is 32-bit VAX-11 condition value
595 0650 1
596 0651 1 ++
597 0652 1 FUNCTIONAL DESCRIPTION:
598 0653 1
599 0654 1 Converts from FORTRAN error number to 32-bit VAX-11 condition value
600 0655 1 complete with proper severity and all other fields set.
601 0656 1 If already a 32-bit condition value (ie GTRU FOR$K ERR_MAX),
602 0657 1 no conversion is done. Instead the FORTRAN error # is FOR$K_NOTFORSPE
603 0658 1 which has a value of 1 and indicates a non-FORTRAN specific error.
604 0659 1
605 0660 1 FORMAL PARAMETERS:
606 0661 1
607 0662 1 FORT_ERR_NO ! Value of FORTRAN error # (0:120) or 32-bit cond value
608 0663 1
609 0664 1 IMPLICIT INPUTS:
610 0665 1
611 0666 1 NONE
612 0667 1
613 0668 1 IMPLICIT OUTPUTS:
614 0669 1
615 0670 1 NONE
616 0671 1
617 0672 1 ROUTINE VALUE:
618 0673 1 COMPLETION CODES:
619 0674 1
620 0675 1 32-bit VAX-11 condition value.
621 0676 1
622 0677 1 SIDE EFFECTS:
623 0678 1
624 0679 1 --
625 0680 1
626 0681 2 BEGIN
627 0682 2 MAP
628 0683 2 FORT_ERR_NO: BLOCK [4, BYTE]; ! Could be a condition value
629 0684 2 LOCAL
630 0685 2 VAX_11_COND_VAL: BLOCK [4, BYTE]; ! 32-bit VAX-11 error condition value
631 0686 2
632 0687 2 ++
633 0688 2 Convert FORTRAN error code to 32-bit VAX-11 error code, unless already
634 0689 2 a 32-bit condition value (some other facility than FOR$ in LH).
635 0690 2 Conversion is done by copying FORTRAN error number to code field,
636 0691 2 setting the severity code to SEVERE, except error 63 (OUTPUT CONVERSION ERROR)
637 0692 2 in which case the severity is set to ERROR.
638 0693 2 Thus the user must do something explicit in order to continue
639 0694 2 for all errors, except 63 (but it has ***s so error flagged).
640 0695 2 Therefore the user will not inadvertently use data which had errors in it.
641 0696 2 setting the facility code to FOR$K_FAC_NO,
642 0697 2 and setting the facility specific bit (ST$&V_FAC_SP).
643 0698 2
644 0699 2 --
645 0700 2 IF .FORT_ERR_NO LEQU FOR$K_MAX_ERR
646 0701 2 THEN
647 0702 2 BEGIN
```

```

: 648      0703 3      VAX_11_COND_VAL = 0;
: 649      0704 4      VAX_11_COND_VAL[ST$V_SEVERITY] = (IF .FORT_ERR_NO EQL FOR$K_OUTCONERR
: 650      0705 4      THEN
: 651      0706 4      ST$K_ERROR
: 652      0707 4      ELSE
: 653      0708 4      ST$K_SEVERE);
: 654      0709      VAX_11_COND_VAL[ST$V_CODE] = .FORT_ERR_NO;
: 655      0710      VAX_11_COND_VAL[ST$V_FAC_SP] = 1;
: 656      0711      VAX_11_COND_VAL[ST$V_FAC_NO] = FOR$K_FAC_NO;
: 657      0712      END
: 658      0713      ELSE
: 659      0714      VAX_11_COND_VAL = .FORT_ERR_NO;
: 660      0715
: 661      0716      RETURN .VAX_11_COND_VAL
: 662      0717      END;

```

! End of COND\_VALUE routine

				0000 00000 COND_VALUE:				
		0000005D	BF	04	AC D1 00002	.WORD	Save nothing	: 0646
					27 1A 0000A	CMPL	FORT_ERR_NO, #93	: 0700
					51 D4 0000C	BGTRU	3\$	
			3F	04	AC D1 0000E	CLRL	VAX_11_COND_VAL	: 0703
					05 12 00012	CMPL	FORT_ERR_NO, #63	: 0704
			50		02 D0 00014	BNEQ	1\$	
					03 11 00017	MOVL	#2, R0	
			50		04 D0 00019	BRB	2\$	
51	03	00			50 F0 0001C	MOVL	#4, R0	
51	0C	03	04		AC F0 00021	INSV	R0, #0, #3, VAX_11_COND_VAL	: 0709
		51	8000		8F A8 00027	INSV	FORT_ERR_NO, #3, #T2, VAX_11_COND_VAL	: 0710
51	0C	10			18 F0 0002C	BISW2	#32768, VAX_11_COND_VAL	: 0711
					04 11 00031	INSV	#24, #16, #T2, VAX_T1_COND_VAL	: 0700
		51	04		AC D0 00033	BRB	4\$	: 0714
		50			51 D0 00037	MOVL	FORT_ERR_NO, VAX_11_COND_VAL	: 0716
					04 0003A	MOVL	VAX_T1_COND_VAL, R0	: 0717
						RET		

; Routine Size: 59 bytes, Routine Base: \_FOR\$CODE + 01C2

```

: 663      0718 1 END
: 664      0719 0 ELUDOM

```

! End of module

## PSECT SUMMARY

Name	Bytes	Attributes
_FOR\$CODE	509	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)



## Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	27	0	581	00:01.0
-\$255\$DUA28:[FORRTL.OBJ]FORLIB.L32;1	711	186	26	52	00:00.6
-\$255\$DUA28:[FORRTL.OBJ]RTLLIB.L32;1	36	0	0	8	00:00.1

## COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:FORSIGNAL/OBJ=OBJ\$:FORSIGNAL MSRC\$:FORSIGNAL/UPDATE=(ENH\$:FORSIGNAL)

: Size: 509 code + 0 data bytes  
: Run Time: 00:15.6  
: Elapsed Time: 00:39.4  
: Lines/CPU Min: 2772  
: Lexemes/CPU-Min: 16368  
: Memory Used: 144 pages  
: Compilation Complete



0183

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY



0184 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY